Handling Class Imbalance In Credit Card Fraud Detection

Tarandeep Singh Mandhiratta
Master of Applied Computing
Wilfrid Laurier University
Waterloo, Canada
mand4710@mylaurier.ca

***Abstract*— Fraud is defined as any hostile conduct intended to defraud the other party of money. Even in underdeveloped nations, the popularity of digital currency or electronic currency is increasing, as is the fraud linked with it. Consumers and banks have lost billions of dollars due to credit card fraud throughout the world. Fraudsters continue to explore new methods and tactics to commit fraud despite the existence of multiple systems to prevent it. To counter these scams, we need a sophisticated fraud detection system that not only identifies the fraud, but also identifies it before it occurs and accurately. Our systems must also be able to learn from previous frauds and adapt to new fraud schemes in the future. Credit card fraud is a rising issue that costs billions of dollars each year throughout the globe. Innovative classification algorithms allow financial institutions to identify fraudulent transactions without interfering with legal transactions or wasting resources on fraud forensics. Unfortunately, there are some significant hazards, such as concept drift and imbalance learning. In this article, two days of European credit card transactions are used to test current state-of-the-art strategies for dealing with class imbalance at the data and algorithm levels. The obtained results are compared to a benchmark for three algorithms that have previously been shown to perform well in fraud detection research: random forest, multi-layer perceptron, and linear support vector machine. When high class imbalance arises, advanced generative sampling methods might possibly fail effectively generalize the minority class, resulting in inferior performance than more traditional class imbalance solutions such as cost-based methods.**

Keywords*— *class imbalance, ADASYN, support vector machine, multi-layer perceptron, random forest

I. Introduction

According to numerous sources, credit card theft results in billions of dollars being stolen every year [1] . Since the number and diversity of online payments grows, so does the multitude of frauds, as it is simpler to mask one's identification and position on the Internet. The developments in technology and availability of access have given criminals new chances to expedite their action plan while maintaining their anonymity. A layer of defense will almost certainly be inadequate to support cardholders, retailers, and issuing institutions from a potential assault. Another layer should be provided to identify these abnormalities in a proactive manner [2]. As a result, there has been a lot of study towards detecting and preventing fraud.

Data-mining methods have become well-established in the last several years. However, because to concerns about privacy, study in this field is severely constrained. Machine learning approaches have been used for fraud detection since the 1990s, with today's algorithms growing more complex. For consumers, organizations, and the financial sector, fraud using debit, credit, and pre - paid cards is a serious and rising problem [3].

Financial institution's use of software to prevent credit card fraud has historically tracked advancements in categorization, clustering, and pattern recognition [4][5]. Most fraud detection systems now use more complex machine learning algorithms that identify and detect fraudulent behaviors in real time and offline, with minimum disruption to legitimate transactions[6][7][8].

In general, fraud detection systems must address several unique problems associated with the task, including extreme dataset unbalance because frauds account for a small percentage of total transactions, evolving distributions due to changing consumer habits, and assessment challenges associated with real-time data processing [9]. Many machine intelligence algorithms, for example, are not built to manage excessively significant variances in class sizes [8], which creates problems when learning from imbalanced datasets. In addition, dynamic changes in the data need strong algorithms that can tolerate idea drift in real customer activities [10].

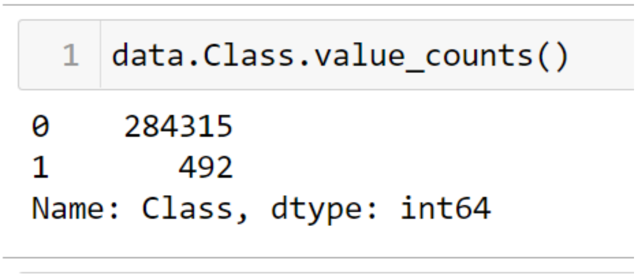
Despite the existence of specialized techniques such as fuzzy inference systems, knowledge-based systems, and outlier detection that can handle large class imbalances, existing research suggests that traditional algorithms can be used successfully if the information is sampled to generate equivalent class sizes [11][12]. Using a clustering method like k-nearest neighbors, the most recent sampling strategies entail constructing fake data. This is advantageous because it allows for the use of a broader variety of standard classification methods, including those that are off-the-shelf, so alleviating algorithmic limits caused by excessive class imbalance[13][14][15]. Not only can fraud recognition capabilities increase as a result of a broader range of possible methods, but development costs can be reduced as a result of less complete dependence on highly technical specialty methodologies, expert systems, and ongoing research into computational methods that directly address class imbalance.

II. Dataset

The credit card fraud dataset utilized in this study is available at Kaggle.com [16], and it covers a selection of electronic European credit card payments done during two days in September 2013, with 492 frauds from a total of 284807 transactions. The dataset is merely supplied as 28 unnamed columns as a consequence of a PCA transformation for reasons of secrecy. There are three named columns as well: Amount, Time, and Class [10][11][12][17][18].

The dataset is heavily skewed, with the positive class (frauds) accounting for just 0.172 percent of all transactions. It only has numerical input variables that have undergone a PCA transformation. We are unable to give the original and basic information about the dataset owing to confidentiality concerns. The major components derived with PCA are features V1, V2,... V28; the only features not changed by PCA were 'Amount' and 'Time'. The transaction Amount is represented by the feature 'Amount,' which may be utilized for dependent cost-sensitive learning. The time lapsed between each purchase and the very first transaction are stored in the feature 'Time'. The return variable is called 'Class,' and it has a value of 1 when there is fraud and 0 when there isn't [16].

It's worth noting that the dataset is heavily skewed toward genuine transactions with the label "0," as seen in Figure 1.



```
data.Class.value_counts()
```

```
0    284315
1       492
Name: Class, dtype: int64
```

Figure 1: Class imbalance in the credit-card fraud dataset

III. IMPLEMENTATION & METHODOLOGY

Credit card fraud is on the rise, plus it comes with several issues, including severe class imbalance and temporal drift. SMOTE (Synthetic Minority Oversampling Technique) and ADASYN (Adaptive Synthetic Sampling Approach) are two state-of-the-art strategies used in this research to address class imbalance. The training dataset consists of about 280000 genuine transactions done in Europe in September 2013. Multi-Layer Perceptron, Support Vector Machine and Random Forest are the machine learning techniques that are examined. According to the result, the best sampling strategy for an unbalanced dataset depends on the model and the dataset being used.

1) *Concept Drift:*

Seasonality, new items and Consumer preferences, as well as shifting fraud attack techniques, all contribute to concept drift in credit card theft. The end result is that the underlying data's statistical features change with time. Current findings have shown that issues can be avoided while still using traditional machine intelligence approaches [10]. An ensemble approach, in which the oldest element is substituted with a new classifier, or a sliding window strategy, in which a classifier is trained on the most recent data, are two instances [10].

For two reasons, however, the issues related with concept drift, along with their remedies, are not discussed in this study. One is that the set of data used is accumulated over just a period of two days, which would not be adequate for concept drift to take place. The other reason is because, as previously stated, research suggests that concept drift for detection techniques may be effectively addressed by using traditional approaches that merely keep a localized temporary memory of learnt properties [10]. To put it another way, once a technique is discovered that works well enough for short durations, i.e. sufficiently short lengths of time when concept drift doesn't really occur, its implementation may be updated to compensate for concept drift. As a result, before applying the fraud detection algorithms discussed in this research to a data stream longer than a few days, they would need to be refined further. These improvements are explored in further depth near the conclusion of the study.

2) *Imbalance Learning:*

Information gain is used as the splitting criteria for learning in standard decision trees like C4.5 and ID3, resulting in rules that are skewed towards the majority. Unbalanced datasets are also a concern for neural networks, support vector machines and k-nearest neighbors, according to research [19][20]. This issue is exacerbated when the two basic classes intersect, as in the Kaggle dataset; most machine - learning algorithms are ill-equipped to deal with both imbalanced and overlapping class distributions [5].

Fortunately, certain algorithms are available that can adjust for class imbalance. Furthermore, there are strategies that may lessen the harmful consequences of these biases at the data and algorithm levels.

3) *Sampling:*

By lowering the size of the classes to approaching equality, sampling techniques are utilized to account for the dataset's imbalance. Oversampling and undersampling, which both utilize a bias to accomplish this goal, are basically comparable and opposing approaches. Instead of merely reproducing the minority class, more advanced algorithms like the adaptive synthetic sampling approach (ADASYN) and the state-of-the-art synthetic minority oversampling technique (SMOTE) construct new data points depending on known samples and their attributes

[5][21]. However, these methods are computationally costly since they depend on assumptions made by the minority class. Specifically, the constructed data is often an interpolation of past data, which may or may not offer a meaningful estimate of whether or not the classes were in actuality balanced. Despite this, sampling methods are more resilient than other approaches to imbalance learning, such as cost-based strategies that punish mistakes differently depending on class, favoring the minority class [15][22].

In any scenario, sampling using adaptive synthetic sampling approach on the dataset is compared to conventional undersampling in previous studies. In Python3.7.9, the library "Imbalance-Learn"[23] was utilized. The validation and testing data are not sampled, therefore the final accuracies given are not skewed. This is reflective of real-life situations in which fraudulent occurrences would have been in the minority. Finally, if relevant, these findings would be examined to cost-based balancing approaches.

4) *Classification:*

To distinguish between fraudulent and genuine transactions, several fraud detection systems employ supervised classification algorithms. Random Forest (RF) outperforms Support vector machine and Neural network when undersampling is used to account for class imbalance in comparable credit-card fraud datasets, according to research [9]. This conclusion is supported by tests on the Kaggle dataset using three types of techniques: neural networks, ensemble methods, and linear approaches. In this study, we look into Multilayer perceptron (MLP), Random Forest, and linear Support vector machine utilizing ADASYN training data.

a) Multi-Layer Perceptron:

A fully connected feed - forward neural network called a multilayer perceptron (MLP) creates a set of outputs from a collection of inputs. A MLP may be viewed of as a deep artificial neural network. It is made up of many perceptrons. They are made up of an input layer that receives input, an output layer that makes a judgment or prediction about the input, and an unspecified number of hidden layers in between that comprise the MLP's real computational engine.

Because each node in the network uses a non-linear activation function, MLPs may categorize data that isn't linearly separable. Standard backpropogation is used to train the network. In this project, MLP is implemented using Python 3.7.9 using "scikit-learn" v1.0.1 [24]. Variant MLP designs are also explored using "Tensorflow" v1.13.1. [25]

b) Random Forest:

Random forests, also known as random decision forests, are an ensemble learning approach for classification, regression, and other problems that works by training a large number of decision trees. Ensemble approaches, rather than relying on a single algorithm, use a number of weak classifiers to get better results. RF is a method that creates a forest out of a large number of decision trees. RF offers the benefits of being efficient on little quantities of information, efficient and resilient on huge datasets, and easy cost-based balancing [16]. The RF algorithm used in the code was created using Python 3.7.9 and "scikit-learn" v1.0.1[24].

c) Support Vector Machine:

Support-vector machines (SVMs) are supervised learning methods that examine information for regression and classification analysis in machine learning. A SVM's goal is to build a hyperplane among data points in space, namely support vectors, so that the samples are isolated by the largest gap feasible. The new data points are classified by which side of the gap they land on. Although there are non-linear approaches for SVM, a binary linear classifier is typically utilized. Using Python 3.7.9 and "scikit-learn" v1.0.1, the SVM method was employed in this project [24]. This SVM solution allows for cost-based balancing to be applied.

5) *Validation:*

It's necessary to keep in mind that a dataset with a lot of imbalance will indicate a high baseline precision by default. For instance, the Kaggle data used in this article has a baseline precision of of 99.827%. That is, when a binary classifier constantly picked the class "no fraud," a maximum precision would be recorded because there are so less fraudulent activities in comparison to valid ones. However, it is evident that the purpose of detecting fraudulent transactions will not be met. As a result, a more thorough method is required: a confusion matrix with False Positive (FP), False Negative (FN), True Positive (TP), and True Negative (TN) values (FN).

	True Fraud	True Legitimate
Predicted Fraud	TP	FP
Predicted Legitimate	FN	TF

Figure 2: Confusion Matrix

Precision, the percentage of occurrences accurately categorized as fraudulent, and recall, the proportion of fraudulent cases correctly classified, may be derived from these rates.

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

The confusion matrix, as well as recall and precision measures, give an in-depth analysis of a fraud detection system's performance [5][26]. Specifically, FP is permitted, but FN is not, since it is preferable for a financial firm to invest resources on a valid alarm rather than overlook a scam entirely. Financial institutions as well as other credit card companies see fraud analytics teams as an essential expense of doing business, but the consequences of missing frauds are considerably more serious, including a loss of consumer confidence and legal implications. In this scenario, the recall measure is critical since every FN lowers the score. Confusion matrices are generated in Python 3.7.9 using "scikit-learn" v1.0.1. [23]

The algorithm performance is further evaluated using the classification report given by scikit-learn v1.0.1. With regard to both f-score and classes, this report includes recall and precision values. This data is useful for assessing the algorithm's FP and FN provision.

Different categorization methods may be more successfully evaluated by measuring the area under the precision-recall (PR) curve (AUPR) rather than focusing just on accuracy [7][10][26]. A comprehensive picture of a classifier's performance may be acquired by evaluating it over the whole range of thresholds. Utilizing "scikit-learn" v1.0.1 with Python 3.7.9, the AUPR is re-formulated. [24]

Finally, the receiving operating characteristics (ROC) curve is added to the AUPR (AUROC). For a binary classifier, the ROC is just the TP rate versus the FP rate when the discriminating threshold is changed [26]. It's comparable to the PR curve in that the larger the area under the curve, the better the performance, but it evaluates an algorithm differently, particularly recall versus fallout, the likelihood of classifying a genuine transaction as fraudulent.

IV. RESULTS

A. *Support Vector Machine*

On the training data upsampled using ADASYN, the precision-recall and ROC curves for various thresholds of the "c" threshold parameter within linear SVM classifier are as follows:

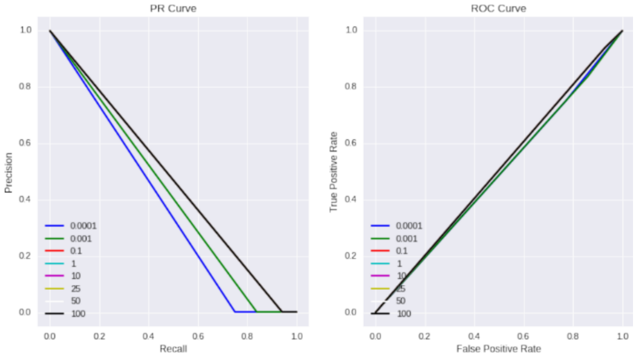


Figure 3: ADASYN, SVM Precision-recall and ROC curve for the c threshold

The consistent negative gradients for all threshold settings reveal the poor performance. Figure 4 contrasts this by demonstrating the same technique and threshold settings, but with raw, unsampled data:

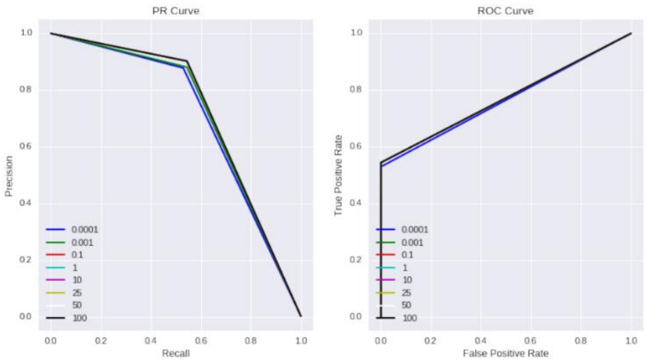


Figure 4: Unsampled, SVM Precision-recall and ROC curve for the c threshold

The training set resampled using ADASYN resulted in inferior performance than the raw unsampled data, as seen in these charts. Nonetheless, both findings imply that a c > 0.1 cutoff value delivers the best outcomes. The accompanying plots are created by applying this finding to the different cost-based approach of class reweighting:

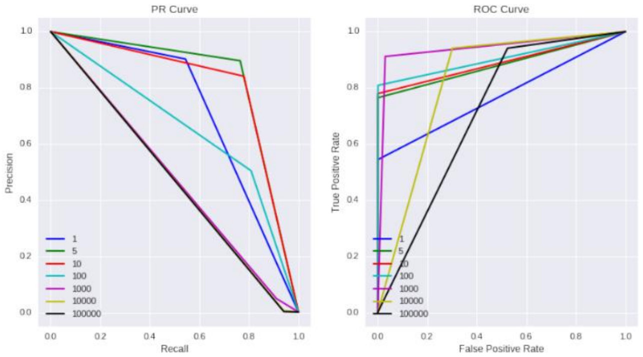


Figure 5: Unsampled, SVM Precision-recall and ROC curve for class_weights, c=1

On unsampled data, a minority class weight of 5:1 over the majority class yields greater performance over both unsampled information without class reweighting and ADASYN sampled dataset, according to the AUPR in Figure. 5. Using the AUROC as a guide, raising the minority class weighting to 1000 would raise the TP rate for a specified FP rate while having no effect on the FP rate.

Having to implement class reweighting to unsampled ADASYN data improves performance over unsampled ADASYN information without class reweighting, however results in worse AUPR and AUROC than the findings shown in Figure. 5, indicating that ADASYN does not contribute much to the classifier:

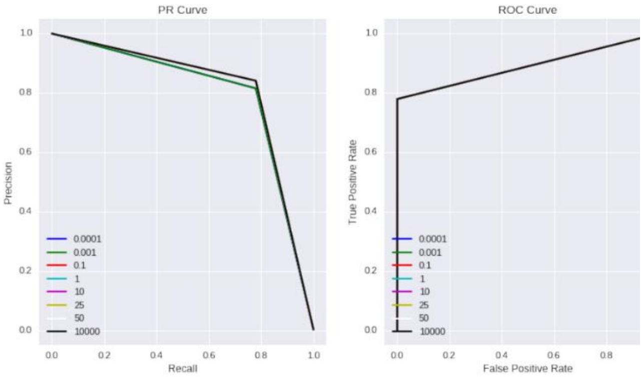


Figure 6: Unsampled, SVM Precision-recall and ROC curve for c threshold, class_weight={1:5,0:1}

Examine the FN rates in the following confusion matrices to see how ADASYN without class reweighting compares to the unsampled data using class reweighting:

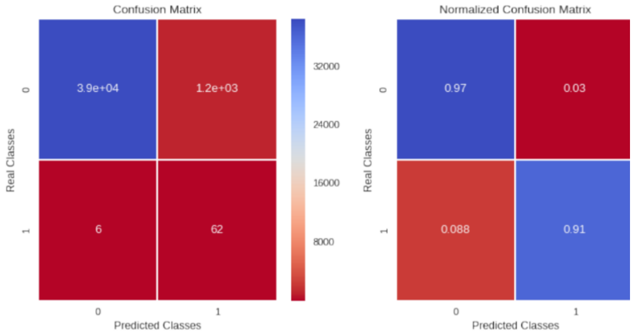


Figure 7: Unsampled, SVM, c=1, class_weight={1:1000,0:1}

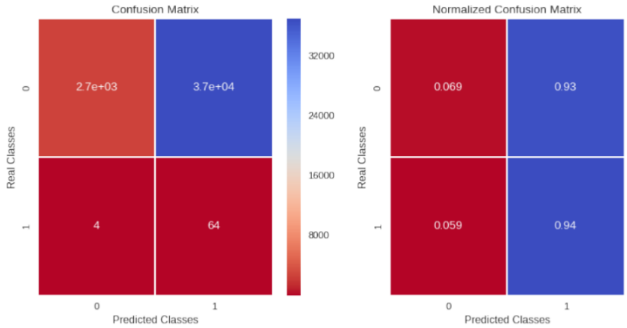


Figure 8: ADASYN, SVM, c=1

Upon first inspection, it seems that the classifier employing ADASYN detects fraudulent transactions more accurately than before when trained using unsampled reweighted data, as seen in Figure. 7. However, the greater minority class recall comes at the expense of a far greater FP rate of 93%. Clearly, the ADASYN-trained classifier is heavily skewed toward the minority group. The performance of SVMs trained on unsampled information with class weighting is superior: 97 percent TN and 91 percent TP. The FN rate is lowered by about 37% as contrasted with training unsampled data in the absence of class reweighting.

B. Random Forest

Even though it is believed that RF has superior performance over NN and SVM [6], the data show different. When training using ADASYN sampled training data, Figure 9 shows very poor performance for almost any choice of the quantity of estimators parameter:

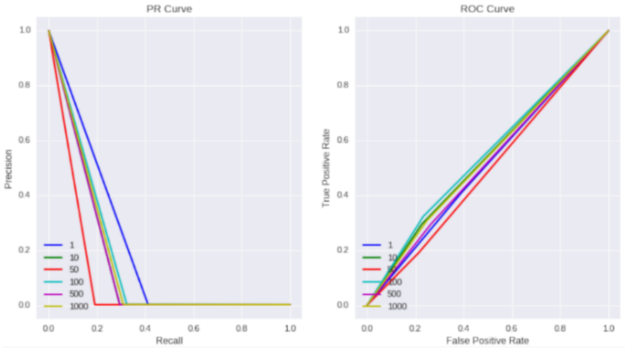


Figure 9: ADASYN, RF Precision-recall and ROC curve for n_estimator

In comparison, when the same procedure is applied to unweighted and unsampled data, the following results are obtained:

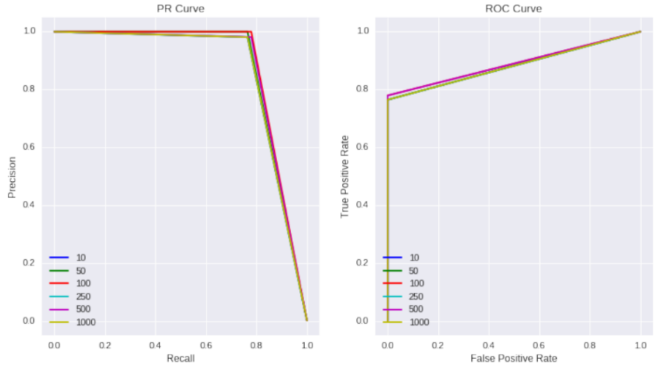


Figure 10: Unweighted & unsampled, RF Precision-recall and ROC curve for n_estimator

Selecting the highest performing parameter for the number of estimators and monitoring AUPR and AUROC for various class reweighting doesn't quite provide as much performance gain in the case of RF as it does in the case of SVM, as shown in Figure. 11 below:

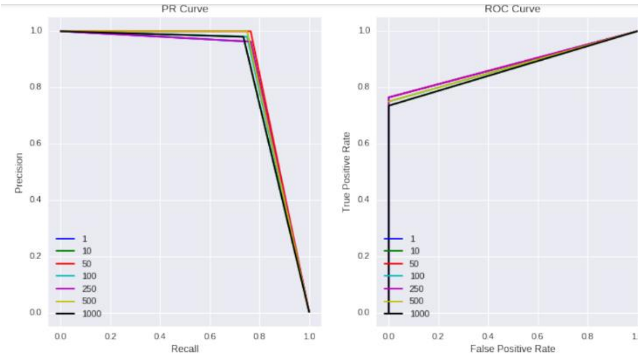


Figure 11: Unsampled, RF Precision-recall and ROC curve for class_weight, 100 estimators

While analyzing the confusion matrices for RF by utilizing ADASYN sampled training data versus unsampled data with class reweighting, it is clear that ADSYN provides much lower results: Both the TP and FP rates are in jeopardy:

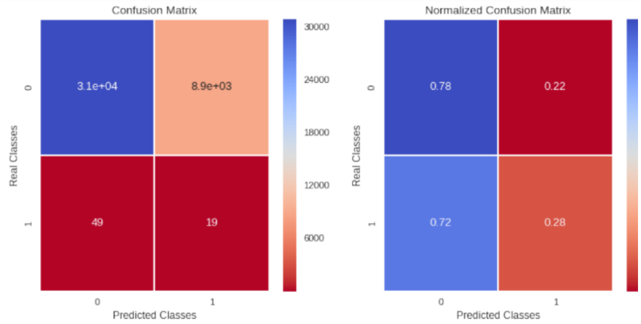


Figure 12: ADASYN, RF confusion matrices, 100 estimators

Because it is skewed towards the majority class, the predictor trained utilizing ADASYN has a poor fraud detection performance. This is intriguing since it contradicts the findings produced using SVM and ADASYN.

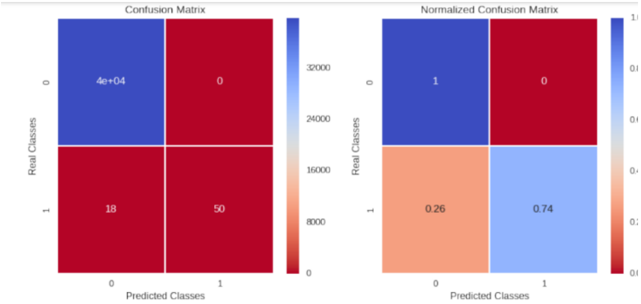


Figure 13: RF confusion matrices, 100 estimators, class_weight={1:10,0:1}

Figure 13 illustrates that the RF on unsampled but reweighted classes gives a flawless FP rate however a TP rate of 74%, which is much lower than SVM.

C. *Multi-Layer Perceptron*

When employing MLP, similar pattern is seen. Using sampled ADASYN training data with stochastic gradient descent (SGD), varying the number of layers demonstrates poor performance throughout all attribute values:

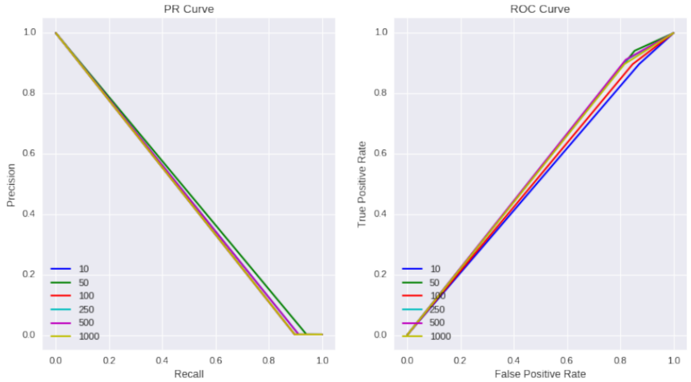


Figure 14: ADASYN using SGD, MLP Precision-recall and ROC curve for n_layers,

A layer size of 50 is optimum as per the AUROC and AUPR, however any layer size will yield either a high TN or TP rate, never both.

This is seen in Figure. 15 below, which was generated using the same approach but with unsampled unweighted data.

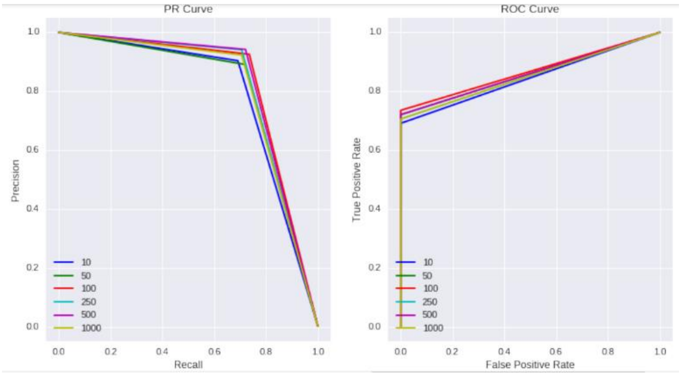


Figure 15: Unsampled & unweighted using SGD, MLP Precision-recall and ROC curve for n_layers.

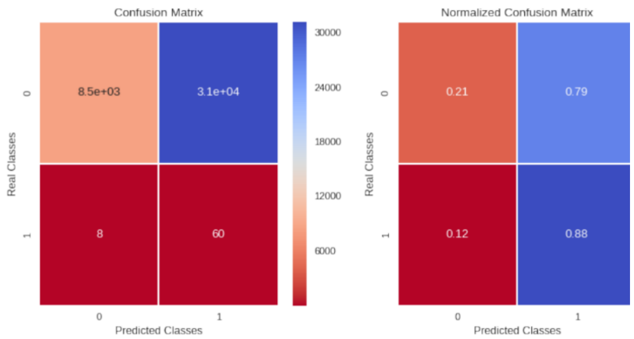


Figure 16: SGD, ADASYN, MLP confusion matrices, 100 layers,

The MLP is clearly crippled when utilizing ADASYN sampled training data. This conclusion is supported by the confusion matrices:

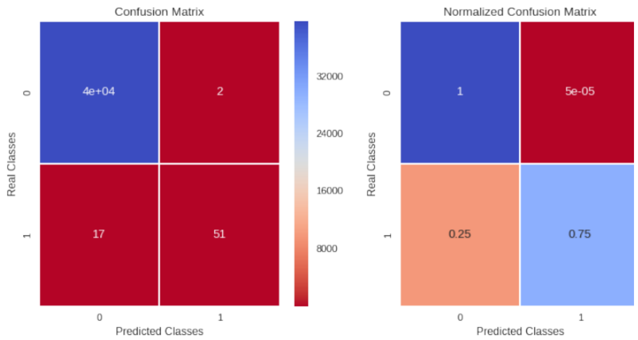


Figure 17: SGD, unweighted & unsampled, MLP confusion matrices, 100 layers

Notably the TP rate in Figure. 16 is rather high, but this is offset by a high FP rate, showing a bias towards the minority class.

According to these findings, an MLP based on unsampled data outperforms ADASYN sampled training data in terms of fraud detection system effectiveness, but it has a substantially higher FN rate than other techniques.

On unsampled and unweighted training data, the efficiency of the MLP developed with SGD is approximately comparable to that of the RF utilizing reweighted classes learned on unsampled data. The difference is a little higher FP rate and just a slightly reduced FN rate.

V. ANALYSIS

The findings demonstrate that the dataset upsampled using ADASYN performed much worse than the unsampled data. Additionally, both the ADASYN sampled and unsampled datasets performed better than the unsampled data utilizing alternative class weighting to favor the miority class. This is in direct opposition to recent studies, which found that ADASYN improved overall fraud detection effectiveness when compared to other methods. [6][21]

The disparity is thought to be caused by two factors.

1) The data collected did not reflect actual fraudulent transactions. This may be accomplished in three steps: fraudulent transactions do not have strong enough distinction when compared to valid transactions, at least in the dataset employed, or that ADASYN failed to develop synthetic samples that captured the underlying feature of fraudulent transactions, Or maybe, the class disparity was just too severe to be mitigated by ADASYN. The positive information acquired utilizing unweighted and unsampled approaches, on the other hand, show that the classes can be distinguished. As a result, it's thought that ADASYN was unable to develop fresh fake samples that were indicative of the data. It's unclear if this is entirely attributable to the huge class disparity or is related to the data's nature.

2) After training with evenly balanced data created by synthetic sampling, the particular techniques of the RF, SVM, and MLP algorithms employed were unable to generalize to substantially unbalanced test and validation data. The increased FP rate while ADASYN is employed supports this.

Furthermore, while SVM equipped with unsampled dataset with class reweighting outperformed MLP trained with unweighted classes and unsampled data in terms of fraud detection, it is suggested that class reweighting be investigated for MLP training. This is due to the fact that MLP outperforms SVM on unweighted and unsampled information. Despite being an uncommon approach, it is believed that even if class reweighting can be applied to the MLP, comparable performance increases will be realized as with SVM.

VI. FUTURE SCOPE

The study on identifying credit card theft has a lot of promise for the future. The genuine factors that may be tracked enabling credit card fraud identification can be known when a record with decrypted fields is published to the public. As a result, credit card firms will be better educated on the most critical elements to consider when forecasting credit card fraud, and their notification systems will be more efficient [27]. Furthermore, the conclusions of this experiment were hampered by the data set's tiny sample size of bogus instances. The algorithms may be taught to generate more precise predictions by utilizing a bigger dataset with a higher number of fraudulent instances. More computational power may be needed to achieve these objectives. To enhance the efficiency of training and validation each algorithm with a bigger, more complicated dataset, it may be necessary to consider employing a Graphical Processing Unit [28][29].

Upsampling methods like ADASYN may be used with traditional class imbalance mitigation approaches like class reweighting for further investigation. Although using both techniques resulted in lower performance than class reweighting alone in in this paper, it is possible that by fixing for the impacts that ADASYN had on the outcomes in this article, utilizing a supplementary method that has been shown to optimize effectiveness might very well result in an optimal solution fraud detection system that can utilize off-the-shelf functionalities of traditional classifiers.

Finally, the findings of this study effort may help determine the optimal method to utilize in other scenarios of skewed data analysis, such as global catastrophe prediction.

VII. CONCLUSIONS

The findings contradict previous research that suggests that upsampling using approaches like ADASYN improves binary classification performance in severely unbalanced datasets. When compared to traditional strategy for coping with class imbalance, like undersampling or even cost-based methods, it is obvious that producing synthetic samples might provide far poorer results. In many circumstances, it may actually harm the classifier and yield results that are worse than ignoring class imbalance altogether.

The upsampled information in the study is examined more closely in order to see and comprehend the properties of the synthetic samples, as well as to determine how representative they are of true fraudulent samples. Whenever the class imbalance is 99.8% in favor of the majority class, it's possible that trying to upsample too much results in a data that, if trained on a classifier, is constantly biased in favor of the minority class, as shown by high FP rates.

When the reliability of the various classifiers utilized is compared, it is evident that linear SVM outperforms MLP and RF. In this scenario, the optimum classification was achieved using unsampled data for training with class reweighting. MLP, which was trained using unsampled and unweighted data, came in second, but with a three-fold higher FN rate than SVM. With class reweighting and unsampled training data, RF came in third.

REFERENCES

[1] Bolton, R. J., & Hand, D. J. (2002). Statistical fraud detection: A review. Statistical science, 235-249.

[2] Leonard, K. J. (1993). Detecting credit card fraud using expert systems. Computers & industrial engineering, 25(1-4), 103-106.

[3] Adewumi, A. O., & Akinyelu, A. A. (2017). A survey of machinelearning and nature-inspired based credit card fraud detection techniques. International Journal of System Assurance Engineering and Management, 8(2), 937-953

[4] Ghosh, S., & Reilly, D. L. (1994, January). Credit card fraud detection with a neural-network. In System Sciences, 1994. Proceedings of the Twenty-Seventh Hawaii International Conference on (Vol. 3, pp. 621-630). IEEE.

[5] Srivastava, A., Kundu, A., Sural, S., & Majumdar, A. (2008). Credit card fraud detection using hidden Markov model. IEEE Transactions on dependable and secure computing, 5(1), 37-48

[6] A. D. Pozzollo, “Adaptive Machine Learning for Credit Card Fraud Detection,” Ph.D. dissertation, Dept. Comp. Sci., Univ. Libre de Bruxelles, Bussels, Belgium, 2015.

[7] L. Delamaire, H. Abdou, and J. Pointon. Credit card fraud and detection techniques: a review. Banks and Bank Systems, 4(2):57–68, 2009.

[8] A. D. Pozzolo, et. al., “Calibrating Probability with Undersampling for Unbalanced Classification” in Symposium on Computational Intelligence and Data Mining (CIDM), 2015 © IEEE. doi: 10.1109/SSCI.2015.33

[9] A. D. Pozzollo, et. al., “Learned lessons in credit card fraud detection from a practitioner perspective”, submitted for publication in Expert Systems with Applications, Feb 2014.

[10] A. D. Pozzollo, “Credit Card Fraud Detection and Concept-Drift Adaptation with Delayed Supervised Information” in International Joint Conference on Neural Networks (IJCNN), 2015 © IEEE. doi: 10.1109/IJCNN.2015.7280527

[11] M Krivko. “A hybrid model for plastic card fraud detection systems”. Expert Systems with Applications, 37(8):6070–6076, 2010.

[12] Y. Sahin, S. Bulkan, and E. Duman. “A cost-sensitive decision tree approach for fraud detection”. Expert Systems with Applications, 40(15):5916–5923, 2013.

[13] H. He and E. A Garcia. “Learning from imbalanced data. Knowledge and Data Engineering”, IEEE Transactions on, 21(9):1263–1284, 2009.

[14] G. Batista, A. Carvalho, and M. Monard. “Applying one-sided selection to unbalanced datasets” in MICAI 2000: Advances in Artificial Intelligence, pages 315–325,2000.

[15] J. Laurikkala. “Improving identification of difficult small classes by balancing class distribution”. Artificial Intelligence in Medicine, pages 63–66, 2001.

[16] Kaggle. (2017, Jan. 12). Credit Card Fraud Detection [Online].Available:https://www.kaggle.com/mlg-ulb/creditcardfraud

[17]https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm

[18] A. D. Pozzollo, “When is undersampling effective in unbalanced classification tasks?” in European Conference on Machine Learning and Principles and Practice of Knowledge Discovery, Porto, Portugal, 2015, pp. 200-215.

[19] S. Visa and A. Ralescu. “Issues in mining imbalanced data sets-a review paper” in Proceedings of the sixteen midwest artificial intelligence and cognitive science conference, pages 67–73. 2005.

[20] I. Mani and I. Zhang. “knn approach to unbalanced data distributions: a case study involving information extraction” in Proceedings of Workshop on Learning from Imbalanced Datasets, 2003.

[21] H. He, et al., “ADASYN: Adaptive synthetic sampling approach for imbalanced learning”. in IEEE International Joint Conference on Neural Networks, pages 1322–1328. IEEE, 2008.

[22] G. M. Weiss and F. Provost. “The effect of class distribution on classifier learning: an empirical study”. Rutgers Univ, 2001.

[23] G. Lemaitre, F. Nogueira, and C. K. Aridas, “Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning,” submitted for publication.

[24] Pedregosa, et. al., “Scikit-learn: Machine Learning in Python” in JMLR 12, pp. 2825-2830, 2011. [Online] Available: http://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html

[25] M. Abadi, et. al., “TensorFlow: Large-scale machine learning on heterogeneous systems”, 2015. [Online] Available: http://download.tensorflow.org/paper/whitepaper2015.pdf

[26] J. Davis and M. Goadrich. “The relationship between precision-recall and roc curves”. in Proceedings of the 23rd international conference on Machine learning, pages 233–240. ACM, 2006.

[27] Şahin, Y. G., & Duman, E. (2011). Detecting credit card fraud by decision trees and support vector machines

[28]Shen, A., Tong, R., & Deng, Y. (2007, June). Application of classification models on credit card fraud detection. In 2007 International conference on service systems and service management (pp. 1-4). IEEE.

[29] Van Hulse, J., Khoshgoftaar, T. M., & Napolitano, A. (2007, June). Experimental perspectives on learning from imbalanced data. In Proceedings of the 24th international conference on Machine learning (pp. 935-942). ACM.